Improving Access to Software Architecture Knowledge An Ontology-based Search Approach

Adriana Maria Figueiredo, Julio C. dos Reis, Marcos A. Rodrigues

Software Development Technologies Lab
Center for Information Technology Renato Archer Campinas, SP, Brazil

Abstract

Every software has an architecture, but in most cases it is poorly documented. That's especially true in virtual community for software development, where creating and updating documents is not a high priority activity and architectural knowledge remain hidden in informal and semi-structured artifacts, such as e-mails, meeting notes and Wikis. We propose a framework that enables the search for software architecture information in the artifacts generated in virtual communities environments. An ontology-based semantic search mechanism is used in the framework to retrieve not only architecture properties, but also the rationale behind decisions taken during the construction of the software. The ontologies defined are also used in the visualization of the search results, providing an interactive environment for users to explore, discover, and analyze the information. The framework is preliminary validated in the Brazilian Public Software Community context.

Keywords: software architecture, ontology, semantic search, software development community

1. Introduction

New methods for retrieving documents based on their meanings are becoming more and more important in many application areas. In this paper, we present a semantic search mechanism that enhances the search for software architecture information in Virtual Communities for Software Development (VCSD). Our proposal is to add semantics to the community artifacts, using ontologies that represent concepts relevant to understand the software. These concepts are the software architecture, architectural design rationale and application domain abstractions.

The methods for information retrieval based on the use of ontologies have emerged as a response to the shortcomings of the model based only on the lexical representation of terms. The shortcomings of this model directly affect the requirements of accuracy and coverage of search engines: documents unrelated to the search are retrieved while the relevant documents are not. These problems occur because the traditional search indexes are based on lexical representation of the documents. These are terms that represent vague information, and sometimes represent more noise than useful and meaningful information to be explored by the search [1]. While the traditional search of document is established primarily on the occurrence of words in documents, the ontology-based semantic search is an information retrieval process that exploits a domain's knowledge, formalized through an ontology [2]. Bonino *et al*. [3] say that the key point to the refining process of a semantic search is the availability of a domain ontology, and the ability to understand the semantic relationships between the ontological concepts. This is important since the searches are very context-dependent due to the various possible meanings of a same word (polysemy phenomenon). A major improvement in the relevance of the results could be achieved knowing exactly what a user means when specifying a search term and with the information description of the contents to be retrieved.

The focus of our work is the retrieval of information that helps to understand the software in a VCSD environment. To achieve this goal, we look for information related to software architecture, a key asset in the software development process, determinant to the success and quality of software systems [4]. Even though, despite its relevance, the architecture documentation is still poor, especially in VCSDs environments, where creating and updating documents is not a primary activity and architectural knowledge remain hidden in casual and semi-structured artifacts, such as e-mails, meeting notes, Wikis, and others. The informality and fragmentation of the shared information make hard to community members to retrieve and consequently to learn and reuse software assets. It is especially difficult to novice participants who may have a hard time to contribute since they do not know why the software is the way it is.

In recent years, a new perspective of software architecture has been envisioned by researchers in the area. In this new view, design decisions are

inherent part of the software architecture, just like components and their relationships. Indeed, the architecture is also the reflection of the decisions taken during its construction. Based on the requirements, the person who architects a software analyzes alternatives and makes decisions that impact the architecture [4]. Therefore, our search mechanism looks for argumentation and reasoning behind the architectural decisions as well.

The new perspective for software architecture has led to a number of research works that promote the capturing and use of architectural decisions in a meaningful way [4, 5]. In [5], López *et al*. propose an approach to retrieve architectural rationale from texts documents. The idea behind this proposal is that ontology's concepts and their relationships provide clues for what to look for in the texts. In this way, a software architecture ontology is used in the generation of ontology-based extraction tools that, in turn, are used to process documents and extract architectural rationale candidates.

We argue that the software architecture ontology is not enough. Considering that a software solves a problem, the software architecture ontology is related to the solution side, contemplating concepts like components, patterns and others. However, a software system is part of a larger business system and a subset of business entities, rules and processes are input for the software construction and part of the architectural rationale as well.

The semantic search mechanism we propose is based on a framework where a software architecture ontology and application domain ontologies model the knowledge of the software development domain. These ontologies are used to index the community's artifacts, as well as to visualize the search results, helping the users to explore, discover, and analyze the information.

The remainder of this paper is organized as follows. In Section 2, we present research works related to ours. In Section 3, we introduce semantic search field and explain the ontologies used. The framework and the semantic search mechanism are detailed in Section 4. A brief description of the community where we are applying and validating the framework is given in Section 5. In Section 6 we discuss implications of our approach, and we conclude this paper in Section 7.

2. Related Work

Our approach is close to the one presented by Lopez *et al*. [5], which uses the Toeska Architecture Ontology and the NFR Design Rationale (NDR) Ontology to represent, recover and explore software architecture and rationale information from text documents. The extraction mechanism is based on Natural Language Processing (NLP) techniques and the main goal of that work is to build a repository of architectural knowledge. Our work has a simpler goal, since we do not aim the construction of a knowledge base, but a search mechanism to help to find this knowledge. To achieve this goal, in addition to the Toeska Architecture Ontology, we propose the use of application domain ontologies to enable the retrieval of all rationale related to the software construction. This architectural knowledge is relevant in a VCSD environment where many projects share the same domain abstractions.

In another related work, Antunes *et al*. [6] use ontologies to promote reuse of software development knowledge. The *Representation Ontology* is used to represent the many artifacts that result from the software development process, such as specification documents, design diagrams, source code and the *Domain Ontology* is used to classify these artifacts. As ours approach, that one also relates software properties with domain concepts. The difference is that while we are concerned with the reuse of software architectural knowledge, in that work the focus is in the reuse of software artifacts.

Similar to our goal to improve information access in a virtual community context, in the work presented in Deng *et al*. [7] the authors propose domain-driven search through the use of meta-queries. Although the modeling of the domain is required in order to create the meta-queries, a meta-query does not represent the knowledge of a domain and therefore provides a solution less versatile.

3. Semantic Search

The objective of optimizing search results has motivated research in the semantic field by incorporating techniques from a variety of other research areas, and implementation of a number of practical systems [8]. Semantic search, as an application of the Semantic Web in the information retrieval field, has shown a significant potential in the function of improving the performance of retrieval. Compared with the traditional search engines that focus on the frequency of word appearance, semantic search engines are more likely to try to understand the meanings hidden in retrieved documents and from users' queries [9]. This kind of mechanism tries to analyze what a user desires during a search in a context through a logical reasoner, enabling better results. According to Kassim & Rahmany [10], the traditional search engines are no longer able to provide precise results due to the huge volume and complexity of the information. The drawbacks of these mechanisms are that they are keyword-based enhanced by link analysis, and not capable to deal with polysemy and synonyms aspects. Therefore, they often return results that do not meet the users' necessities. Semantic search has become an alternative to

overcome the deficiencies of such traditional mechanisms. Different from them, semantic search mechanisms extend the scope of traditional information retrieval (IR) paradigms from mere document retrieval to entity and knowledge retrieval, improving the conventional IR methods by looking at a different perspective, *i.e.* the meaning of words [8].

Search with semantic characteristics demands the mechanism to be based in a knowledge model about the domain, *i.e.*, the knowledge must be computationally represented so that the machine can interpret it. There are several architectural proposals for semantic search solutions. The authors in [2, 18, 19, 20 and 21] have made an extensive revision of the main proposals of semantic search solutions in the literature. They describe open research questions and necessary investigations, as well as similarities, goals, applications, methodologies and technologies involved in the various proposals. Guha *et al.* [9] argue that the semantic search has mainly two goals: the first is to improve and increase the traditional search results and the second is to use the understanding of the search term denotation to improve the significance of the traditional search. They describe how the improvement can be achieved, saying that there are three main issues to be addressed: (1) denotation: it is necessary to determine the concept denoted by the search term; (2) what to present: to determine what relevant data to return; and (3) how to present: it is necessary to format the data (search results) properly in the presentation.

In the Web context, unlike traditional search mechanisms, the semantic search mechanisms store semantic information about resources on the Web and are able to solve complex queries. It may consider the context where the resource is targeted. A semantic search mechanism in the Web context may integrate Semantic Web technologies with search engines to improve the search results retrieved by the current search engines. It goes a step further towards to the next search generation in the Web. The need to retrieve information semantically enriched gave rise to an increasing interest in research on ontologies, giving rise to the called ontology-based semantic search mechanisms.

3.1. Ontology-based Semantic Search

A search mechanism with semantic features needs to interpret a knowledge model about a domain and an ontology is an approach to represent this knowledge [16, 9, 3 and 17]. A wide adopted definition for ontology in Computer Science is done by Gruber [15]: ontology can be understood as a specification of a conceptualization which provides descriptions about knowledge.

By the fact that an ontology defines relevant concepts in a domain (things and their properties) and the semantic relationships between these concepts, it supports the processing of resources based on a coherent understanding of the content and not in the physical structure of resources or in a syntactic feature. Thus, the use of these artifacts can be some kind of semantic (world knowledge) for the machine [9]. According to Hyvönen *et al.* [11] the following benefits can be achieved using semantically richer ontologies: (1) they can be used to describe the domain knowledge and the terminology of the application in greater detail (*e.g.* relations between classes in different views); (2) ontologies can be used to create more accurate semantic annotations in Web resources in terms of domain knowledge; (3) with the help of ontologies, users can express queries more accurately and possibly without ambiguity, which leads to better rates of accuracy and coverage of the search; and (4) through ontological class definition and inferences mechanisms, such as property inheritance, instance-level metadata can be enriched semantically.

3.2. Software Architecture Ontology and Domain Ontology

The software architecture we defined is based on the Toeska Architecture Ontology [5]. A partial view of this ontology is depicted in Figure 1 and the main concepts are: a software component, which may be a component, a system or a subsystem, satisfies requirements and architectural alternatives, uses technologies and is built in a project, responsibility of a community. A software system is based on well-established patterns, which in a VCSD environment, have a crucial role since they provide a common vocabulary and frame of reference to ease architectural knowledge sharing among developers, who are geographically distant and with distinct levels of experience and skills [4].

Unlike the approach adopted in López *et al.* [5], where the Softgoal Interdependency Graphs (SIG) method is used to represent the extracted rationale that is then validated and integrated in a repository, in our approach the rationale is identified, ontology instances are generated and an entry in the index points to the artifact with the rationale text.

As mentioned previously, business entities, rules and processes are input for the software construction and hence needed in the mechanism to gather architectural rationale. Figure 2 presents an ontology with some concepts and relationships of the learning domain. Some abstractions from this domain are modeled in a software system.

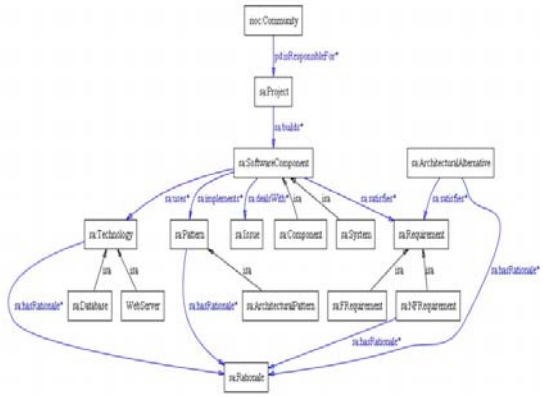

Figure 1. Software Architecture Ontology

To illustrate the use of both ontologies to capture rationale decisions, we present two excerpts with decisions related to the software architecture from the Amadeus Community [12].

(1)

According to the alternatives presented ….. the development team …. judges as the best alternative option of using an ISAPI Filter so that IIS can redirect requests Java, due to utilization of resources already in place such as Apache Tomcat ……..

(2)

During the development of the List of Concepts (a special type of learning object), we decided to build its content dynamically, i.e. the concepts and details of each are read from a text file in CSV (comma-separated values). This allows the contents of this application to be easily developed by the teacher … This feature facilitates the participation of the teacher who has a fundamental role in the design of educational artifacts ……..

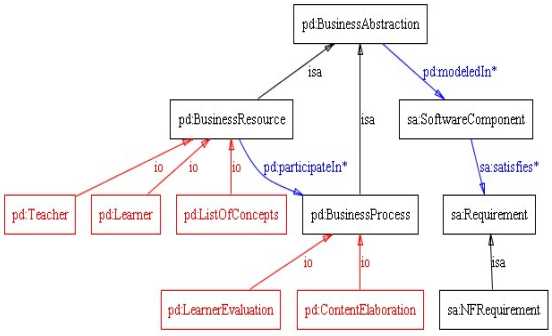

Figure 2. Learning Ontology and its relationship with Software Architecture Ontology

With the use of NLP techniques, IIS, Java, ISAPI, Apache Tomcat are identified as technologies (1), and "list of concepts" and "teacher" are business abstractions in (2). The non-functional requirements are reuse (utilization) in (1) and usability (easily developed) in (2). The presence of these entities in a sentence plus the relationships in the ontologies helps to identify rationale in the texts. The information units extracted at this point are candidates that must be validated by a software specialist. In our approach we use natural language process algorithms to content analysis and heuristic techniques to detect rationale.

4. Proposed Framework

The setup of the framework's ontologies, the semantic index and the search mechanism are detailed in the next sub-sections.

4.1. Setup

The ontology's concepts presented in the previous section represent classes in a domain. To populate the ontologies with theirs instances, we adopt a semi-automatic approach for the initial load. To populate the software architecture ontology, we first look for community's artifacts (documents, FAQ, Wiki, emails, *etc.*) from different application domains (education, health and others) in well-known software development forges (SourceForge[1] , Savannah[2] , BerliOS[3]). In the next step, all the content retrieved is processed to remove stopwords and then to extract the most frequent words. These extracted words are candidates that, in the final step of the setup process, are analyzed by a software specialist who, if that is the case, creates the ontology's instances. The definition of the domain ontology must be done by specialists in the area, but a start point for this task is the list of most frequent terms (not related to the programming language) extracted from the source code.

After the ontologies setup, the framework is ready to interact with the VCSD system and this is done through the use of a Web service. By means of this interface, as new artifacts are produced in the community, its content and address (URL) are sent to the framework in order to populate the semantic index. The artifact's content is processed by a POS tagging software and nouns, verbs and adverbs are selected. These selected words are confronted to the ontologies and also to values of the datatype properties of the ontology instances. In this way, it is not just the individual's name used to classify the artifacts. If a word from an artifact matches some value of an individual datatype property, then a new entry is inserted in the semantic index, relating the individual with the artifact identification (its URL). Figure 3 illustrates the processes described above.

In order to detect the rationale the following heuristic is applied:

[1] http://sourceforge.net/

[2] http://savannah.gnu.org

[3] http://www.berlios.de/

for each *"artifact's content"*
if exists *"non-functional requirements terms"* **and** *"terms that indicate decision"*
then "suggest a rationale";

With this heuristic we look for rationale that is related to non-functional requirements and, in the presence of words like, "judge", "decide", "facilitate", and others, we suggest that a rationale maybe exist in that artifact.

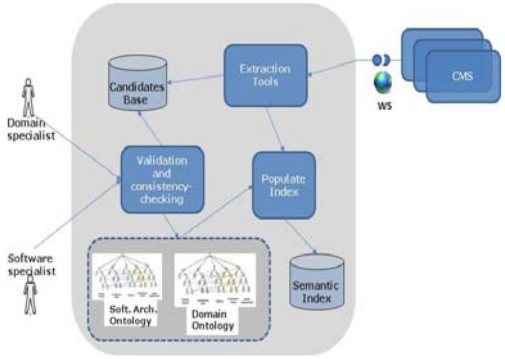

Figure 1. Framework Setup

4.2. The Search Mechanism

The search mechanism must pay special attention to hide the complexity of semantic search from end users while at the same time, make it easy and efficient to use. On the other side, it is interesting to take advantage of the underlying model to make the results more meaningful to users. Bearing this in mind, we propose a search mechanism based on the software architecture and a domain ontology. The user can visualize these ontologies in hierarchical or graph-based way, what allows the user to see different relationships besides the hierarchical one (Figure 4). The tab (hierarchical or graph-based) allows users to choose one visualization type or other. In the Figure 4, as an example, the domain ontology is presented hierarchically while the architecture ontology is presented as a graph.

This approach enables users to search for information from two perspectives: the technical and the domain perspective. User without much knowledge in the technological issues may explore the knowledge first using the domain ontology with high-level requirements or abstractions. More technically users may go through the technologies as the software components, or design patterns using the architecture ontology.

The artifacts are retrieved by browsing one ontology (domain or architecture). As an instance of the ontology is selected, the search results are exhibited in the right side, as depicted in Figure 4.

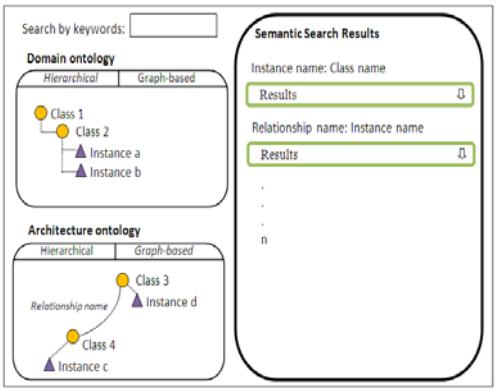

Figure 4. Semantic Search Interface

The first results presented are those related to the instance chosen. Following as a list, the results of ontological instances related to the selected instance in the ontology are presented. One of these results can be a point to a rationale text related to the concept chosen (see Figure 5). The results are retrieved based on the index created that connects the instances to the artifact. Each set of search results (links to the artifacts) is organized into a dropbox that may hide the results. The user is able to open and close the dropboxes. The results retrieved from the index that points out to artifacts are organized based on the instances reached. For all instances reached by the possible navigation in the ontology related to that instance selected, a result box is created. The next section presents a possible evaluation for the proposed approach.

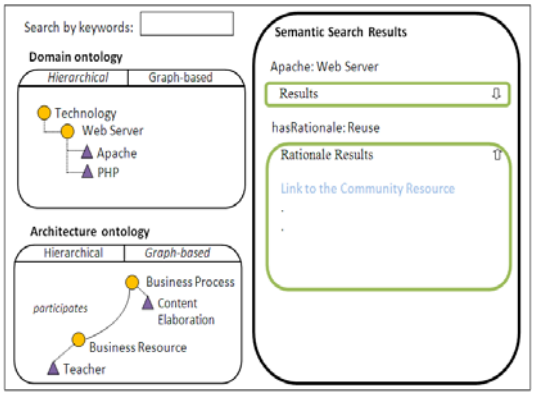

Figure 5. Search Results

5. Case Study

We plan to validate the proposed framework through a case study in the Brazilian Public Software community (SPB) [13]. SPB is government initiative which aims to promote a sustainable free-software business model, integrating the society into a new model of technological knowledge production. Nowadays, SPB hosts more than 40 communities,

developing software solutions to domains such as health, education, and others. In an assessment realized in the SPB Portal regarding software architecture documentation [14], it was detected that only one-third of the communities have some kind of documentation for the software architecture. When available, in many cases all the information provided is just the list of the technologies used. This lack of architectural knowledge weakens the replication of good software solutions in different municipalities. Brazil is a large country, and the adaptation of good public software solutions to local requirements by local developers is one of the objectives of the SPB project. In this way, a semantic search mechanism, as the one we propose, is expected to help people to understand the software in development and strengthens the chances of replication.

The setup of the framework for validation in the SPB communities includes the selection of an application domain, selection of a group of communities related to the domain and the definition of the ontology. The challenges to be addressed are: (1) domain specialists to collaborate in the ontology are geographically distant; (2) conflicts and inconsistencies regarding the understanding of the domain concepts; (3) technical knowledge by the community's members to create the ontologies in a collaborative matter; and also (4) tools supporting such activity.

The case study will be applied to one of the selected communities and the goal is to check how easy is to understand the software with the semantic search mechanism. A group of software developers without prior contact to the software and the community will read the community artifacts. Half of the group will use the semantic search mechanism and the other half will browse the community. A questionnaire will be applied to evaluate how easy was to each group to understand the software.

6. Discussion

In a VCSD environment, architectural assets from one community can be reused by others. The discovery and successful recovery of the architectural knowledge is essential to the possible reuse and interoperation of solutions as well. To achieve that, approaches and techniques are necessary to empower people from different communities and technical background to find different types of assets and the reasoning behind their choice. Toward that, it is needed the understanding and mapping of the data from the semantic point of view to achieve accurate, useful and reusable information. From this perspective the knowledge could be shared more effectively between the communities.

In the proposed approach, the domain ontology enables users to have a better vision of the software requirements and their relation to the domain under study to the community. In addition, the domain and architectural ontologies are complementary views and together provide a complete access to the architectural knowledge. The access to parts of the knowledge extracted from the content according to semantic relationships lead to a better understanding of the architectural knowledge. The access to the project's rationale improves the quality for future decisions in the definition of new software architectures to other contexts or communities.

The proposed approach to populate the ontologies is semi-automatic and that is not desirable since it increases the workload for the person who holds the role of ontology specialist in the community. But, nevertheless, the fully automatic generation of ontologies without human assistance is to date a challenging research issue.

7. Conclusion and Future Work

In this paper we have described a framework that empowers virtual communities for software development with a semantic search mechanism to retrieve architectural knowledge. The mechanism is based on ontologies for software architecture concepts and domain specific abstractions.

The contribution pursued by this work was to provide a lightweight mechanism that can be easily adopted by the community and useful especially for new developers who need to understand the software properties and the decisions taken during its construction. The user interface based on the ontologies provides an interactive environment for users to explore, discover, and analyze information based on a semantically understanding of the data. It also allows users from different backgrounds to explore the information from a perspective more appropriated to his experience. In future work, we plan to extend the search for different types of rationales, not only those related to non-functional requirements as well as to design improved visualization tool to deal with a better exploration of the knowledge by means of the ontologies.

8. References

[1] Botero, S. W., "Extração de relações semânticas via análise de correlação de termos em documentos", Dissertation in Portuguese, FEEC/Unicamp, 2008.

[2] Mangold, C., "A survey and classification of semantic search approaches", Int. J. Metadata, Semantics and Ontology, Vol. 2, No. 1, pp.23–34, 2007.

[3] Bonino, D.; Corno, F.; Farinetti, L.; Bosca, A., "Ontology Driven Semantic Search", WSEAS

Transaction on Information Science and Application, Issue 6, Vol. 1, pp. 1597-1605, 2004.

[4] Babar M. A., Dingsøyr T., Lago P., van Vilet H., "Software Architecture Knowledge Management - Theory and Practice", Springer-Verlag , 2009.

[5] López C., Codocedoa V., Astudilloa H., Cysneiros L. M., "Bridging the gap between software architecture rationale formalisms and actual architecture documents: An ontology-driven approach", Science of Computer Programming, doi:10.1016/j.scico.2010.06.009m 2010.

[6] Antunes B., Seco N., Gomes P., "Using Ontologies for Software Development Knowledge Reuse", Lecture Notes in Computer Science, Volume 4874/2007, 357-368, DOI: 10.1007/978-3-540-77002-2_30, 2007.

[7] Deng Y., Devarakonda M., Rajamani N., Zadrozny W., "Improving Information Access for a Community of Practice Using Business Process as Context", IEEE 24th International Conference on Data Engineering, 2008.

[8] Wang W., Barnaghi P. M., Bargiela A., "Search with Meanings:An Overview of Semantic Search Systems". International Journal of Communications of SIWN, Vol. 3, pp. 76-82, 2008.

[9] Guha, R., McCool, R., Miller, E., "Semantic Search". Proceedings of the 12th international conference on World Wide Web. pp. 700-709, Hungary, 2003.

[10] Kassim, J. M., Rahmany, M., "Introduction to Semantic Search Engine". In International Conference on Electrical Engineering and Informatics. Selangor, Malaysia. Vol. 02, pp. 380-386, 2009.

[11] Hyvönen, E., Saarela, S., Viljanen, K., "Application of Ontology Techniques to View-Based Semantic Search and Browsing. The Semantic Web: Research and Applications", Lecture Notes in Computer Science, Vol 3053, pp. 92-106, 2004.

[12] Amadeus – A Learning Management System http://www.softwarepublico.gov.br/spb/ver-comunidade?community_id=9677539. (Access date: 4 August 2011).

[13] Brazilian Public Software Portal: http://www.softwarepublico.gov.br/spb/. (Access date: 4 August 2011).

[14] Figueiredo A., "Arquitetura de Software no SPB - Avaliação da Documentação", unpublished, technical report in Portuguese, TRT01181010 , 2010.

[15] Gruber, T. R., "A translation approach to portable ontologies". Knowledge Acquisition, Vol. 5, Num. 2, pp. 199-220, 1993.

[16] Heflin, J., Hendler, J., "Searching the web with SHOE", Artificial Intelligence for Web Search. In AAAI Workshop, AAAI Press, pp. 35–40, 2000.

[17] Fang, W.-D., Zhang, L., Wang, Y.-X., & Dong, S.-B., "Toward a Semantic Search Engine based on Ontologies", In Proc. of the 4th International Conference on Machine Learning and Cybernetics. pp. 1913-1918, 2005.

[18] Wei, W., Barnaghi, P. M.,Bargiela, A., "The Anatomy and Design of A Semantic Search Engine", Tech. rep., School of Computer Science, University of Nottingham Malaysia Campus, 2007.

[19] Hildebrand, M., Ossenbruggen J., and van Hardman, L. , "An analysis of search-based user interaction on the semantic Web". Report, CWI, Amsterdam, Holland, 2007.

[20] Hoang, H., Tjoa, A. "The State of the Art of Ontology-based Query Systems: A Comparison of Existing Approaches", In: Proceedings of the IEEE International Conference on Computing & Informatics, Paper-Nr. 99, 2006.

[21] Dong, H., Hussain, F. K., Chang, E., "A Survey in Semantic Search Technologies", Second IEEE International Conference on Digital Ecosystems and Technologies, Thailand, pp. 403-408, 2008.

9. Acknowledgements

This paper is based on work funded in part by FINEP/MCT grants in the context of Brazilian Public Software Framework Project, contract number #01.08.0604.00, from 29/12/2008.

